5.10 Fast Fourier Transform (FFT)

One wonders if the DFT can be computed faster: Does another computational procedure an **algorithm** exist that can compute the same quantity, but more efficiently. We could seek methods that reduce the constant of proportionality, but do not change the DFT's complexity $O(N^2)$. Here, we have something more dramatic in mind: Can the computations be restructured so that a **smaller** complexity results?

In 1965, IBM researcher Jim Cooley and Princeton faculty member John Tukey developed what is now known as the Fast Fourier Transform (FFT). It is an algorithm for computing that DFT that has order $O(N\log N)$ **for certain length inputs**. Now when the length of data doubles, the spectral computational time will not quadruple as with the DFT algorithm; instead, it approximately doubles. Later research showed that no algorithm for computing the DFT could have a smaller complexity than the FFT. Surprisingly, historical work has shown that Gauss2l in the early nineteenth century developed the same algorithm, but did not publish it! After the FFT's rediscovery, not only was the computation of a signal's spectrum greatly speeded, but also the added feature of **algorithm** meant that computations had flexibility not available to analog implementations.

Exercise 5.9.1

Before developing the FFT, let's try to appreciate the algorithm's impact. Suppose a short-length transform takes 1 ms. We want to calculate a transform of a signal that is 10 times longer. Compare how much longer a straightforward implementation of the DFT would take in comparison to an FFT, both of which compute exactly the same quantity.

To derive the FFT, we assume that the signal's duration is a power of two: $N=2^L$. Consider what happens to the even-numbered and odd-numbered elements of the sequence in the DFT calculation.

$$\begin{aligned} S(k) &= s(0) + s(2)e^{(-j)\frac{2\pi 2k}{N}} + \ldots + s(N-2)e^{(-j)\frac{2\pi(N-2)k}{N}} \\ &\quad + s(1)e^{(-j)\frac{2\pi k}{N}} + s(3)e^{(-j)\frac{2\pi(2+1)k}{N}} + \cdots + s\left(N-1\right)e^{(-j)\frac{2\pi(N-(2-1))k}{N}} \\ &= \left[s(0) + s(2)e^{(-j)\frac{2\pi k}{\frac{N}{2}}} + \ldots + s(N-2)e^{(-j)\frac{2\pi\left(\frac{N}{2}-1\right)k}{\frac{N}{2}}} \right] \\ &\quad + \left[s(1) + s(3)e^{(-j)\frac{2}{\pi}} + \ldots + s(N-1)e^{(-j)\frac{2\pi\left(\frac{N}{2}-1\right)}{\frac{N}{2}}} \right] e\frac{-(j2\pi k)}{N} \end{aligned}$$

(5.39)

Each term in square brackets has the form of a

$$\frac{N}{2} - length$$